Chapter 4: General Product Installation Information

This chapter provides general information you need to know before you start installing products. It discusses:

- the three ways to install **UPS** products (**UPD**, **UPP** and **FTP**)
- how to register your node to download products from `KITS`
- how to determine where **UPD** installs products on your system
- how to declare a product instance to a database manually
- some questions that can come up during an installation
- post-installation procedures required for some products
- how to handle networking restrictions at off-site locations

Installing products into AFS space is not covered in this chapter; see section 9.3 *Installing Products into AFS Space*.

4.1 Installation Methods for UPS Products

There are three ways to access products from a **UPS** product distribution node: using **UPD**, **FTP** or **UPP** (which is actually a layer on top of **UPD**). Each method is described briefly below, and then in more detail in the following chapters. Information on troubleshooting a problematic product installation is provided in Chapter 10: *Troubleshooting UPS Product Installations*.

4.1.1 UPD

The **UPD** product includes the **`upd install`** command for installing products. This is the most widely-used product installation method on machines running **UPS/UPD**. Chapter 6: *Installing Products Using UPD* is dedicated to describing this process. Installation parameters are set in the local node's **UPD** configuration. The aspects of the configuration that you as a product installer need to be aware of are described in section 4.3 *What You Need to Know before Modifying Your UPD Configuration*; the **UPD** configuration is described in detail in Chapter 32: *The UPD Configuration File*.

The **`upd install`** command performs the following functions:

- retrieves the specified product instance, and by default its dependencies, from a distribution node
- unwinds the product (if transferred in tar format) and installs it, and by default its dependencies, on the user node according to the node's **UPD** configuration
- declares the product, and by default its dependencies, to the database specified in the node's **UPD** configuration
- either resolves dependencies or prints to screen the commands you will need to issue in order to do so

4.1.2 UPP

UPP is a layer on top of **UPD** that can be used to perform a variety of tasks, as described in Chapter 33: *The UPP Subscription File*. Regarding product installation, it can be configured to run **`upd install`** for specified products under specified conditions. Dependencies of the specified products are updated automatically, as well, so that the integrity of the products is maintained. **UPP** can also be given instructions to run the necessary **`ups declare`** commands to resolve dependencies when a product installation finishes. **UPP** can be run manually, or it can be automated using a tool like **cron**. Chapter 7: *Installing Products Using UPP* illustrates how to use it to install products.

4.1.3 FTP

Anonymous **FTP** is available on *fnkits*, and may be available on other **UPS** product distribution nodes. **FTP** does not take advantage of the **UPD** configuration. It can be used only to retrieve products; it is left to the installer to unwind and declare them. Furthermore, if the table file and/or the `ups` directory is (are) not included the tar file, it (they) must be retrieved separately. Chapter 8: *Installing Products using FTP* is describes using **FTP** to install products.

FTP is not recommended for installations into the usual product area; **UPD** is designed and configured specifically for that and should be used instead. **FTP** is more suited to product installations into non-standard locations on your node, e.g., into your own area for use just by you.

On *fnkits*, **FTP** is most useful for off-site users who want to download FermiTools products, which are located under the `/pub` directory. You do not need to be a registered user to obtain the FermiTools products.[1]

4.2 User Node Registration for KITS

In order to download most products from the `KITS` database, the machine you're using must be registered with *fnkits.fnal.gov.* All machines in the `fnal.gov` domain are automatically registered. Off-site machines need to register using the **Product Distribution Platform Registration Request** form at `http://www.fnal.gov/cd/forms/upd_registration.html`.

If you only want to download FermiTools products, which are located under the `/pub` directory in `KITS`, you do not need to be using a registered node. FermiTools are made available to the general public.

4.3 What You Need to Know before Modifying Your UPD Configuration

When you install a product using **UPD** (or **UPP**), the installation parameters are controlled by the **UPD** configuration. The **UPS** configuration file for the database you're using points to a **UPD** configuration file. These configuration files described in Chapter 31: *The UPS Configuration File* and Chapter 32: *The UPD Configuration File*. The **UPD** configuration file typically consists of one or more stanzas, each of which:

- identifies certain product instances, products or groups of products
- specifies a database on the local system in which to declare a product matching the identifier
- specifies locations on the local system in which **UPD** is to put a matched product and its related files
- (optionally) lists actions for **UPS/UPD** to perform either just before or just after declaring the product

4.3.1 Location of UPD Configuration File

The Default UPD Configuration File

The **UPD** configuration file is stored as:

---

1. All machines in the `fnal.gov` domain are automatically registered to download products from `KITS`. Off-site machines need to register using the **Product Distribution Platform Registration Request** form at `http://www.fnal.gov/cd/forms/upd_registration.html` .

`${UPD_USERCODE_DIR}/updconfig`

where the keyword UPD_USERCODE_DIR is set in the **UPS** configuration file. It tells you the location of the database containing the **UPD** configuration file. When **UPD** gets setup, the read-only variable ${UPD_USERCODE_DIR} gets defined and set to the same value as the keyword. (The read-only variable ${UPD_USERCODE_DB} also gets defined and set to the database directory containing ${UPD_USERCODE_DIR}). To find the value of UPD_USERCODE_DIR, you can list the **UPS** configuration file, e.g.,:

```
% less $PRODUCTS/.upsfiles/dbconfig
```

or you can first setup **UPD**, and the request the variable value, e.g.,:

```
% echo $UPD_USERCODE_DIR
```

or

```
% env | grep UPD
```

Overriding the Default UPD Configuration

If your system is set up with multiple **UPS** databases configured to point to different **UPD** configurations, you can choose to specify a database on the **`upd install`** command line pointing to a **UPD** configuration file other than the default. First, verify that the database you specify points to the **UPD** configuration you want. To find out, run the command:

```
% ups list -z <database> -K UPD_USERCODE_DIR
```

Note that if this command returns empty quotes, it means the database specifies *no* configuration file. In this case the default **UPD** configuration will not be overridden.

4.3.2 Where Products Get Declared

The keyword UPS_THIS_DB, set in the **UPD** configuration file, identifies the database into which **UPS** declares the product (i.e., the directory that **UPD** specifies in the **`ups declare -z <database>`** option). This keyword may be set differently in different stanzas, thereby causing different products to be declared in different databases.

4.3.3 Where Products Get Installed

For organizational reasons it is usually preferable to have **UPD** configured to install all the **UPS** products for a database in one area. In the **UPS** configuration file, typically the keyword PROD_DIR_PREFIX gets set to the product root directory prefix under which the products reside. The **UPD**

configuration file then defines product root directory locations in terms of PROD_DIR_PREFIX. The quantities you need to be aware of within the **UPD** configuration file are:

UPS_PROD_DIR The product root directory. The **`upd install`** command runs the **`ups declare`** command and uses this value as the argument to the **`-r`** option. It is usually defined relative to PROD_DIR_PREFIX.

UNWIND_PROD_DIR The absolute path to directory where products get unwound. In most cases, it's the product root directory (in terms of read-only variables: `${PROD_DIR_PREFIX}/${UPS_PROD_DIR}`), however in AFS and some NFS mounting configurations, products are often unwound and installed in different locations (see section 9.3 *Installing Products into AFS Space*).

You should not specify the product location in the **`upd install`** command unless you want to override the default.

4.4 Declaring an Instance Manually

A product instance must exist on the system before it can be declared to a **UPS** database[1]. Product declaration is done with the **`ups declare`** command. Declaring a product instance makes it known to **UPS**, and therefore retrievable within the **UPS** framework. Normally products are installed on user nodes using the **`upd install`** command which, in addition to downloading and installing the product, runs **`ups declare`** to make the initial declaration of the product to the local **UPS** database. If you use **FTP** to download a product, then you'll need to declare it manually. Refer to Chapter 8: *Installing Products using FTP* for details about installing with **FTP**.

If you use **`upd install`** and you have more than one database, refer to section 6.3 *How UPD Selects the Database* to see how **UPD** determines the database for the declaration.

1. At least a rudimentary root directory hierarchy for the product, its table file directory and table file must exist before declaration.

4.4.1 The ups declare Command

Before declaring, make sure the product is unwound into in its final location. Also make sure that you've downloaded the table file and installed it in an appropriate directory. For an initial declaration you must specify at a minimum: the product name, product version, product root directory, flavor and table file name[1].

The full command description and option list is in the reference section 23.5 *ups declare*. Here we show commonly used command options (see the notes regarding **`-z`**, **`-U`** and **`-M`** which follow):

```
% ups declare <product> <version> -r /path/to/prod/root/dir/ \
  -f <flavor> [-z /path/to/database] [-U /path/to/ups/dir] \
  [-m <table_name>.table] [-M /path/to/table/file/dir] \
  [<chainFlag>]
```

1) If the database is not specified using **`-z`**, **UPS** declares the product into the first listed database in $PRODUCTS (see section 27.1 *Database Selection Algorithm* for more information).

2) If the product's `ups` directory tar file was unwound in the default location ($<PRODUCT>_DIR/ups), then **`-U /path/to/ups/dir`** is not needed. If the `ups` directory is located elsewhere (or named differently), this specification must be included. If specified as a *relative* path, it is taken as relative to the product root directory.

1. Two exceptions: (1) if the product consists only of a table file that sets up a list of dependencies, there is no product root directory; and (2) if the product has no table file (very rare) then there is no table file name.

3) If the product's table file was placed in either of the two default locations (under `/path/to/database/<product>/` or in the product's `ups` directory), then **`-M /path/to/table/file/dir`** is not needed. Only use the **`-M`** option if you have moved the table file to a separate location where **UPS** won't otherwise find it. If specified as a *relative* path, it is taken as relative to the product root directory. See section 29.4 *Determination of ups Directory and Table File Locations* for details on how **UPS** finds the table file.

Unless the product you're declaring has no table file (true for very few products), make sure its location gets declared properly, either explicitly or by default. Otherwise, users will need to specify its name and location on the command line every time they want to run or operate on the product. If it is neither declared nor specified on the command line, **UPS/UPD** assumes there is no table file.

You can opt to declare a chain to the product instance at this time or in a later declaration. To declare a chain, include the appropriate chain flag in the command (see section 2.3.5 *Chains* for a listing).

4.4.2 Examples

For more examples see the reference section 23.5 *ups declare*.

Declaration of New Product to Non-default Database

The following command shows a fairly typical product declaration. We'll install a product called **histo** v4_0 onto a SunOS+5 node. We assume the product instance's `ups` directory is maintained under its product root directory, and that it contains the table file. We include the **`-z`** option to indicate that we want to override the default database selection. This is the first instance of this product to be declared to this database, therefore the **`ups declare`** command automatically creates the appropriate product directory under the specified database:

```
% ups declare histo v4_0 -f SunOS+5 -m histo.table -z $MY_DB -r\
  /path/to/products/SunOS+5/histo/v4_0
```

We can run a **`ups list -l`** command to see all the declaration information (include **`-a`** because it's not yet declared current):

```
% ups list -alz $MY_DB histo

   DATABASE=/path/to/ups_database/declared
           Product=histo   Version=v4_0    Flavor=SunOS+5
                   Qualifiers=""   Chain=""
                   Declared="1998-04-17 22.08.30 GMT"
                   Declarer="aheavey"
                   Modified="1998-04-17 22.08.30 GMT"
```

```
                Modifier="aheavey"
                Home=/path/to/products/SunOS+5/histo/v4_0
                No Compile Directive
                Authorized, Nodes=*
                UPS_Dir="ups"
                Table_Dir=""
                Table_File="v4_0.table"
                Archive_File=""
                Description=""
                Action=setup
                        prodDir()
                        setupEnv()

   addalias(histo,${UPS_PROD_DIR}/bin/histo)

   addalias(hsdir,${UPS_PROD_DIR}/bin/hsdir)

   envSet(HISTO_INC,${UPS_PROD_DIR}/include)
```

Declaration of Additional Instance of a Product

In the following example we declare an additional instance of **histo**, of the same version, but for the flavor IRIX+5. Again the table file resides under the product root directory's `ups` subdirectory, and we override the default database. This time we declare it with the chain "test" (**`-t`**):

```
% ups declare histo v4_0 -tf IRIX+5 -m histo.table -z $MY_DB -r\
  /path/to/products/IRIX+5/histo/v4_0
```

Running a **`ups list -a`** to see what the database now contains for this product, we find:

```
% ups list -az $MY_DB histo

   DATABASE=/path/to/ups_database/declared
           Product=histo   Version=v4_0    Flavor=SunOS+5
                   Qualifiers=""   Chain=""

           Product=histo   Version=v4_0    Flavor=IRIX+5
                   Qualifiers=""   Chain=test
```

Declaration with Table File Located in Database

Depending on your configuration, you may want the table file to reside in the product's subdirectory under the database (e.g., `$PRODUCTS/<product>/<table_file>`).

A table file for the product must be placed in this location before the instance is declared to the database. Therefore, if you are declaring the first instance of a product to the database, you need to manually create the product directory under the database and copy the table file into it before declaring the instance.

You still do not need to specify the table file location (`-M` option) on the `ups declare` command line; **UPS** will find it here.

4.5 Installation FAQ

4.5.1 What File Permissions Get Set?

Product files get downloaded and installed with the same permissions that they have on the distribution node, minus the **umask** set in your login files. We recommend that you set your **umask** to `002` before installing any products to ensure that you don't remove the group write access for table files.

4.5.2 You're Ready to Install: Should you Declare Qualifiers?

If a product instance is declared with one or more qualifiers on the distribution node, you can choose whether you want to declare it on your system with or without them. If you don't need the qualifiers in order to distinguish between different copies of the same product, it's usually easiest to declare products without them. Otherwise, users must enter the qualifiers on the command line exactly as they appear in the product declaration each time they want to setup that product instance or perform other **UPS** operations on it. The files that **UPS** uses to manage each product allow comment lines (see section 28.1 in Chapter 28: *Information Storage Format in Database and Configuration Files*); this provides a way of recording qualifier information if you choose not to declare the qualifiers explicitly.

4.5.3 What if an Install Gets Interrupted?

Normally **UPD** deletes the installed portions of a product when an installation process gets interrupted, and it doesn't declare the pieces that failed to install. Therefore, you generally don't need to worry about cleaning up before reattempting the installation. Just issue the install command again, the same way as you did the first time.

However, if you interrupted the process for some reason (e.g., you saw it was running out of space), then you'll need to remove by hand the piece that was being installed at the time of the interruption. How will you know? Reattempt the install, and if you get a message similar to this:

```
directory /a/b/c already exists, will not overwrite.
```

then you'll need to remove the specified directory/file(s).

4.5.4 What if a Product was Installed under a Different Name?

Giving a product a new name upon installation can cause problems in dependency trees. This practice is not supported, and is certainly not encouraged, but it can be made to work. If you have a product that needs to find, for example, $MYPROD_DIR, but **myprod** has been installed on your system with a different name, e.g., **fermi_myprod**, then you may need to edit the table file (described in Chapter 36: *Table Files*). Normally product installers never need to touch the table file, but this is an exception. If the provided table file for **myprod** was written by a developer who has no knowledge of the name change on your system, the table file probably contains:

```
ACTION=SETUP
```

```
prodDir()
```

where `prodDir()` instructs the **`setup`** command to set the variable $<PRODUCT>_DIR (see 23.1.4 *More Detailed Description* under section 23.1 *setup*). On your system then, the variable $FERMI_MYPROD_DIR will get set, but $MYPROD_DIR won't. To ensure that you also get the variable $MYPROD_DIR, edit the table file and under ACTION=SETUP add the function:

```
envSet(MYPROD_DIR, ${UPS_PROD_DIR})
```

4.6 Post-Installation Procedures

Some products require that you perform supplementary steps during or after the installation process, for example copying files to other locations or creating needed files or directories. The product's `INSTALL_NOTE` file should contain any instructions for completing the installation. Commonly required actions on the installer's part include configuring and/or tailoring the product instance.

4.6.1 Configuring a Product

Post-installation procedures that can be completely automated are typically collected together such that the command **`ups configure`** executes them. This command gets executed by default, as necessary, when the product instance is declared. Otherwise, you can run **`ups configure`** manually at any time after declaration to configure the product instance.

The configuration may involve creating links to the product root directory from other areas (see section 9.1 *Installing Products that Require Special Privileges*). If the area is not identical for each node (i.e., same path but separate areas) accessing the **UPS** database in which the product instance has been declared, then you will need to run the **`ups configure`** command manually on each node that mounts a unique area. If you are not sure whether you need to configure a product instance on each node, look through the configuration steps in the table file under ACTION=CONFIGURE to see what they do.

4.6.2 Tailoring a Product

Tailoring is the aspect of the product implementation that requires input from the product installer (e.g., specifying the location of hardware devices for a software driver package). If the product requires tailoring, a file is usually

supplied in the format of an interactive executable (script or compiled binary), and it is run via the **UPS** command **`ups tailor`**. You must explicitly tailor the product instance using **`ups tailor`**; tailoring is *not* performed automatically.

Tailoring is generally allowed on any node of a cluster, however we strongly recommend that you perform any node-specific tailoring from that node, or flavor-specific tailoring from a node of that flavor to avoid mismatches.

4.7 Networking Restrictions at your Site

Some off-site locations may impose networking restrictions which can interfere with **UPD**.

4.7.1 Proxying Webserver

If all web traffic is channeled through a proxying webserver at your site, you need to provide the URL of this server to **UPD**. Since **UPD** commands go through a web server, they will fail otherwise (the error message will indicate either "Destination unreachable" or "Timeout"). Look at your web browser configuration to find out what proxy you're using. In **Netscape** this would be under **EDIT/PREFERENCES.../ADVANCED/PROXIES**. Set the variable http_proxy (lower case) to the URL of your server, e.g., (for C shell):

```
% setenv http_proxy "http://some.host.name:8000"
```

4.7.2 Firewall for Incoming TCP Connections

If your site firewalls incoming TCP connections, but allows outgoing ones, you'll need to set the FTP_PASSIVE variable to the value 1, e.g., (for C shell):

```
% setenv FTP_PASSIVE 1
```

This will make **UPD** use "passive mode" **FTP** transfers.